NOISE & VIBRATION Research Group

KU Leuven Department of Mechanical Engineering

KU LEUVEN

Mechanical Engineering · Departement Werktuigkunde

Internal report

KU Leuven in-house multibody research code

Identifier: IR_2014_3

Date: 28/04/2014

Author: Frank Naets

Promotor: Wim Desmet

1. Introduction

This research code is initiated within the Multibody research group by launching a master thesis to create the general architecture which will be the basis for future extension. The goal of the thesis is to create a generic, Matlab-based framework to allows researchers to easily implement, validate and benchmark new formulations for bodies, forces, and joints, but also to implement new model order reduction techniques in an environment capable of simulating flexible multibody systems. The master thesis is performed by Martijn Vermaut under supervision of post-doctoral researcher Frank Naets and Prof. Wim Desmet.

This report lists the functionality that is contained within the research code. This list should not be considered exhaustive as the research code is actively being developed and new functionality is added on a daily basis. Because of this, a listing of the code functionalities and its high-level description is chosen here in favor of a compiled list of available functions and routines.

2. Description of the functionalities

2.1 Bodies

The multibody research code currently contains 5 body definitions. All body definitions have a full description of their inertial and internal forces, as well as their partial derivatives w.r.t. the position, velocity, and acceleration coordinates of the body. The flexible bodies are based on a finite element mesh of which the mass and stiffness matrices are computed using MSC Nastran.

- Ground body: A body of which the position and orientation is kept fixed.
- Point mass body: A body that only has translational degrees of freedom and whose orientation is kept fixed.
- Rigid body based on generalized coordinates, whose center of gravity does not have to coincide with the reference frame.
- Flexible body based on a floating frame of reference formulation. The body formulation uses a reduction space based on the dynamic eigenmodes of the body.
- Flexible body based on a generalized component modes synthesis formulation. The body formulation uses a reduction space based on the dynamic eigenmodes of the body.

2.2 Forces

The multibody research code contains several force definitions. All of these forces have 5 outputs defined:

- F: The force vector expressed in cartesian coordinates (a 6x1 vector containing force and torque).
- B: A mapping matrix to map the cartesian force to the coordinates of the body it is applied to.
- K_t: The derivative of B*F w.r.t. the position coordinates of the body it is applied to.
- C_t: The derivative of B*F w.r.t. the velocity coordinates of the body it is applied to.
- M_t: The derivative of B*F w.r.t. the acceleration coordinates of the body it is applied to. Currently no force definition has a non-trivial implementation for this M_t.

The force definitions that are currently implemented are:

- An aerodynamic force: a force that allows the use to define 6 different frontal areas and 6x6 different aerodynamic coefficients.
- A gravity force: a force that applies a gravitational acceleration to a body. The gravitational acceleration can be entered as an input during the creation of the force.
- An external user-defined input force: a force that allows a user to define an InputData object and use it as a time-dependent force/torque acting on a user-defined frame along/around a user-defined axis.

- A 6-dof spring force : a force that allows the user to define a 6x6 stiffness matrix between two frames.
- A one-axial spring force that uses the implementation fo the 6-dof spring force, but allows for a simpler user interface.
- A 6-dof viscous damping force: a force that alloews the user to define a 6x6 viscous damping matrix between two frames.
- A one-axial viscous damping force that uses the implementation fo the 6-dof viscous damping force, but allows for a simpler user interface.
- A position dependent force: a force that allows a user to define 6x6 position dependent InputData functions that are then applied as a force between two frames. E.g. choosing these functions as linear functions would result in the same forces as the 6-dof spring force.
- A velocity dependent force: a force that allows a user to define 6x6 velocity dependent InputData functions that are then applied as a force between two frames. E.g. choosing these functions as linear functions would result in the same forces as the 6-dof viscous damping force.

2.3 Joints

The multibody research code contains several joint definitions, each with its own constraint residual and jacobian matrix. The jacobian matrices can be computed using an analytical formula or numerically. The available joints are:

- A spherical joint. A joint that fixes the position of two frames.
- A revolute joint. A joint that fixes the position of two frames and makes an axis on each frame colinear with one another.
- A bracket joint. A joint that allows the user which constrints have to be added. The possibilities are fixing the position of two points along a specified axis and forcing an axis on each frame to be perpendicular to one another.
- A slider joint.
- A universal joint.
- A kinematic driver. A joint that takes an InputData object and enforces the function value as a function of time.
- A joint to fix a point to the ground.
- A joint to apply angle parametrisation constraints. An example of this constraint is the constraint that the euler parameters that are used to represent the orientation fo a body, have to form a unit vector.

2.4 Frame Class

An object of the frame class allows developer of the code to uniformely define an interface with a body, independent of the body definition that is used. It allows researchers to define joint and force definitions without having to know the exact body definition beforehand.

Depending on the type of frame, different updating algorithms will be invoked. Note that not all bodies have definitions for all frame types. The implemented frames are:

- Reference frame: a frame that is attached to the reference frame of the body.
- Center Of Gravity frame: a frame that is attached to the center of gravity of the body.
- Fixed frame: a frame that has a fixed position w.r.t. the reference frame.
- Finite Element frame: a frame that is attached to the FE msh (either a node or RBE) of the body.

2.5 InputData Class

An object of the InputData class allows developers to define continuous functions in Matlab without using arrays of data. The interface of each InputData class is the same, so implementation that use continuous function can be made independent of the actual function that is used. The InputData makes the function value as well as its first and second derivatives available to be used. These function can be used as a function of time, but also as a function of position, velocity, or anything else. Current implemented functions include:

- Constant function.
- Linear function.
- Quadratic function.
- Cubic function.
- Polynomial function which fits an n-degree polynomial through the given data.
- Spline function which computes cubic splines through the given data.
- Sinusoidal function.
- Random function.
- Piecewise function which allows users to combine different functions together.

2.6 Solvers

The main solver of the multibody research code is the generalized- α solver with a fixed time step.

There is also a GMP solver that reduces a given system to a GMP reduced system and solves the reduced system using a generalized-α algorithm.

2.7 Others

- A small library of rotation matrices, global and local spin rotation matrix. It also includes their first and second time derivatives, as well as their first and second partial derivatives.
- A set of tools to let Matlab communicate with MSC Nastran in order to compute the mesh stiffness and mass matrices. It also includes a function to parse a BDF file that was used as an input to compute these matrices.
- A set of post-processing tools ranging from plotting the degrees of freedom to displaying a 3d animation with color-coded stresses or displacements or anuthing else a developer might think of.
- Stress vector and strain vector computation for linear solid-elements. There is also a function to convert the stress vectors to von Mises stresses.
- A function to compte the closest orthogonal rotation matrix to a given non-orthogonal matrix and its numerical partial derivatives.